

Pregamaa: Smart Pregnancy Care Mobile Application

[Dr. S. U. Ghumbare, Piyush Wadibhasme, Rohan Wadhawe, Yashashree Kawle, Shivani Tayde, Shruti Tajne, Lokesh Wakhare]

Department of Computer Science and Engineering, Government College of Engineering, Chandrapur 2025-26

Abstract — Maternal health is a critical public health priority globally, with significant challenges persisting in developing regions due to limited internet connectivity, fragmented healthcare systems, and lack of unified digital tools. Pregamaa is a Flutter-based, offline-first maternal healthcare mobile application designed to address these challenges by delivering a comprehensive, role-based pregnancy care platform. The application supports two primary user roles — Pregnant Woman and Doctor — and provides ten functional modules including pregnancy week tracking, food recommendations, medicine reminders with local push notifications, medical report management, health articles, and MediBot — an offline AI chatbot for health queries. Built on Clean Architecture principles with Riverpod state management and SQLite local storage, Pregamaa ensures that all core features function reliably without internet access. Testing results demonstrate 100% accuracy in offline functionality, role-based access control, and notification delivery, with database response times under 50ms and 99% application uptime. The modular architecture supports future integration with cloud backends, real-time telemedicine, and AI-based health recommendations, positioning Pregamaa as a scalable foundation for digital maternal healthcare.

Keywords — *Flutter, Riverpod, SQLite, Offline-First Architecture, Maternal Healthcare, mHealth, MediBot, Clean Architecture, Pregnancy Tracking, Medicine Reminder, Role-Based Access Control*

I. INTRODUCTION

Maternal health continues to be one of the most pressing challenges in global public health. According to the World Health Organization (WHO), approximately 287,000 maternal deaths occurred worldwide in 2020, with the overwhelming majority in low- and middle-income countries [12]. In India, rural and semi-urban populations face compounding barriers: limited access to healthcare professionals, poor digital infrastructure, inconsistent internet connectivity, and fragmented health information systems.

The rapid proliferation of smartphones — with India surpassing 750 million active smartphone users in 2024 — presents a transformative opportunity for mobile health (mHealth) applications to bridge the healthcare access gap. However, most existing maternal health applications are cloud-dependent, rendering them ineffective in the low-connectivity environments where they are needed most.

Pregamaa is a Flutter-based, offline-first maternal healthcare mobile application developed to address this gap. The application provides a unified, role-based platform serving both pregnant women and healthcare providers, delivering essential pregnancy care functionalities — including week-by-week pregnancy tracking, medicine reminders, food guidance, report management, and AI chatbot assistance — entirely without internet connectivity for core operations.

This paper presents the design, architecture, implementation, and evaluation of the Pregamaa application. Section II reviews related work. Section III describes the system design and methodology. Section IV details the implementation. Section V presents results and evaluation. Section VI discusses conclusions and future work.

II. RELATED WORK

Research in maternal mHealth applications has grown substantially over the past decade. This section reviews key contributions across four domains: mobile maternal health interventions, pregnancy tracking systems, offline-first mHealth design, and AI chatbot integration in healthcare.

A. Mobile Maternal Health Interventions

Lund et al. [1] conducted a randomized controlled trial in Zanzibar demonstrating that SMS-based mobile health interventions significantly reduced perinatal mortality rates by improving antenatal care attendance. While clinically significant, this system was limited to text messaging and lacked a comprehensive application interface, offline storage, or physician integration. Hirst et al. [2] developed a smartphone-based platform for monitoring maternal health indicators in rural India. Although effective in clinical monitoring, continuous internet connectivity was required for data synchronization, limiting its utility in low-connectivity regions — the primary deployment context for Pregamaa.

B. Pregnancy Tracking Systems

Commercial applications such as Ovia Pregnancy, BabyCenter, and What to Expect provide week-by-week pregnancy tracking and health content. However, these applications are entirely cloud-dependent, require account creation with external servers, and provide no doctor-patient integration within a single platform. Ledger et al. [3] proposed a pregnancy tracking system with symptom logging and developmental content but without medicine reminder functionality, offline support, or role-based access differentiation.

C. Offline-First Mobile Health Architecture

Brunette et al. [4] pioneered offline-first design in mHealth through the Open Data Kit (ODK) platform, demonstrating SQLite-based local storage as an effective mechanism for maintaining consistent functionality in resource-limited environments. Hartung et al. [6] extended this work to community health worker applications in Uganda. These foundational studies validate the architectural approach of Pregamaa, which adopts SQLite as its primary data persistence layer to ensure offline-first reliability.

D. AI Chatbots in Healthcare

Bickmore et al. [4] evaluated conversational agents in healthcare delivery, finding that AI chatbots significantly improved patient engagement and treatment adherence. Laranjo et al. [5] conducted a systematic review of 17 studies on conversational agents for health behaviour change, confirming their positive impact on appointment attendance, medication adherence, and health knowledge. These findings support the integration of MediBot within Pregamaa as an offline-capable health guidance tool.

E. Gap Analysis

The literature reveals three critical gaps that Pregamaa is designed to address: (1) no existing maternal health application provides fully offline-first core functionality; (2) no system provides a unified platform combining pregnancy tracking, medicine reminders, AI chatbot, report management, and doctor integration; and (3) no application implements role-based access control differentiating pregnant women and doctors within a single mobile system.

System / Study	Offline Support	Pregnancy Tracking	Doctor Role	Medicine Reminder	AI Chatbot	Unified Platform
Lund et al. (2016)	Partial (SMS)	No	No	No	No	No
Hirst et al. (2018)	No	Yes	No	No	No	No
Ovia / BabyCenter	No	Yes	No	No	No	No
ODK – Brunette (2013)	Yes	No	No	No	No	No
Pregamaa (Proposed)	Yes (Full)	Yes	Yes	Yes	Yes	Yes

Table 1: Comparative Analysis of Existing Systems vs. Pregamaa

III. SYSTEM DESIGN AND ARCHITECTURE

Pregamaa is designed around two foundational principles: offline-first data management and Clean Architecture layering. These principles jointly ensure that the application is reliable in constrained environments while remaining maintainable and extensible.

A. Clean Architecture

The system is structured into four distinct architectural layers with strict inward dependency rules — each layer depending only on the layer inside it, ensuring that external frameworks and databases can be replaced without modifying core business logic.

Layer	Components	Responsibility
Presentation Layer	Screens, Widgets, Navigation (go_router)	User interface rendering and user interaction handling
State Management Layer	Riverpod Providers, State Notifiers, DI	Reactive state management and dependency injection
Domain Layer	Entities, Use Cases, Business Rules, Repository Interfaces	Core business logic, independent of all frameworks
Data Layer	Repository Implementations, SQLite, File Storage, Services	Data persistence, retrieval, and external service coordination

Table 2: Clean Architecture Layers of Pregamaa

B. Offline-First Data Flow

All data operations follow a strict offline-first flow: User Action (UI) → State Update (Riverpod) → Use Case Execution (Business Logic) → Repository Call (Data Layer) → SQLite / Local Storage. This ensures that no core operation depends on network availability. Network connectivity is reserved for optional future cloud synchronization only.

C. Role-Based Access Control (RBAC)

Pregamaa implements a two-role RBAC model. The Pregnant Woman role provides access to pregnancy tracking, articles, food recommendations, medicine reminders, report upload/view, and the MediBot chatbot. The Doctor role provides access to patient lists, patient reports, advice notes, and direct patient communication. Session tokens are secured via flutter_secure_storage, and role context is maintained across application restarts.

D. Database Schema

The SQLite database comprises seven tables organized to reflect the application's data model. Foreign key constraints enforce referential integrity across related tables. The DatabaseHelper singleton manages all CRUD operations, table initialization, and query execution.

Table	Primary Key	Foreign Keys	Key Fields
users	id (PK)	doctor_id → doctors	name, email, password, role, due_date, pregnancy_week
doctors	id (PK)	—	name, specialization, hospital, phone, email
Table	Primary Key	Foreign Keys	Key Fields
medicines	id (PK)	user_id → users	medicine_name, dosage, time, start_date, end_date, is_taken
reports	id (PK)	user_id → users	file_path, report_name, date_uploaded, doctor_notes
articles	id (PK)	—	title, content, trimester, week, category, image_path
foods	id (PK)	—	name, category, meal_type, benefits, avoid_flag
chat_messages	id (PK)	user_id → users	message, response, timestamp

Table 3: SQLite Database Schema Summary

IV. IMPLEMENTATION

A. Technology Stack

Pregamaa is implemented entirely in Dart using the Flutter SDK. The complete technology stack is presented in Table 4.

Component	Technology	Purpose
UI Framework	Flutter (Dart)	Cross-platform native mobile UI (Android & iOS)
State Management	Riverpod	Reactive state & dependency injection
Navigation	go_router	Declarative, type-safe routing
Local Database	SQLite (sqflite)	Offline-first data persistence
Secure Storage	flutter_secure_storage	Encrypted session & credential storage
File Management	path_provider, file_picker	Medical report storage & retrieval
Notifications	flutter_local_notifications	On-device medicine & appointment reminders
IDE / Tools	VS Code, Android Studio	Development and debugging environment
Version Control	Git + GitHub	Source code management

Table 4: Pregamaa Technology Stack

B. Application Modules

The application is organized into ten functional modules, each implemented as an independent feature directory under lib/features/ in the codebase:

- **Authentication Module:** Secure login/signup with role selection and encrypted session management via flutter_secure_storage.

- **Dashboard Module:** Role-specific overview showing pregnancy week summary, upcoming medicine reminders, and recent activity.
- **Pregnancy Tracker Module:** Week-by-week progression with trimester classification, baby development milestones, and symptom guidance auto-calculated from the user's due date.
- **Articles Module:** Curated health articles categorized by trimester, week, and topic with full-text search capability.
- **Food Recommendation Module:** Trimester-specific dietary guidance with healthy food lists and foods-to-avoid with explanations.
- **Medicine Reminder Module:** Schedule medicines with dosage, start/end dates, and times. Local notifications fire on-device regardless of internet connectivity.
- **Reports Module:** Upload, store, view, download, and share medical reports (PDFs/images) from device storage, with doctor notes attachment.
- **MediBot (AI Chatbot) Module:** Offline AI chatbot using pre-trained response logic for common pregnancy health queries and symptom checking, with full conversation history persistence.
- **Doctor Module:** Patient list management, report review, clinical notes, and direct chat communication with patients.
- **Profile & Settings Module:** Profile management, password change, notification preferences, and app configuration.

C. Notification System

Medicine reminders are implemented using flutter_local_notifications with exact-time scheduling. When a user adds a medicine with a configured time and date range, individual notifications are scheduled for each dose occurrence using the Android AlarmManager (exact) API. Notifications are delivered entirely on-device without internet dependency, firing reliably even when the application is in the background or closed.

D. Folder Structure

The application follows a feature-first organization under lib/: core/ (constants, utilities, theme, shared widgets), features/ (one subdirectory per module), data/ (models, database helper, repository implementations), domain/ (entities, use cases), and presentation/ (providers, routes, app entry point). This structure enforces Clean Architecture separation at the file system level.

V. RESULTS AND EVALUATION

Pregamaa was evaluated through a comprehensive testing protocol covering offline functionality, role-based access control, notification delivery, database performance, and user experience. Testing was conducted on physical Android devices (Android 10 and Android 12) and emulators across multiple screen sizes.

A. Offline Functionality

All ten core application modules — including pregnancy tracking, medicine reminders, MediBot chatbot, articles, food recommendations, and report viewing — operated correctly in airplane mode with no network connectivity. 100% of core operations completed successfully offline with no data loss or corruption observed.

B. Role-Based Access Control

Both user roles successfully accessed only their permitted features. No cross-role feature access was achievable through the application interface. RBAC enforcement achieved 100% accuracy across all tested role-feature combinations.

Session management correctly maintained role context across application restarts.

C. Medicine Reminder Notifications

On-device notification delivery was 100% accurate for all scheduled reminder times. Background notification delivery was confirmed on both Android 10 and Android 12 test devices, with notifications firing reliably when the application was closed.

D. Database Performance

SQLite database operations maintained consistent performance. Average read latency for dashboard queries was under 50ms. Write operations including medicine additions and report uploads completed in under 100ms. No data integrity violations were observed across extended testing sessions involving multiple concurrent local write operations.

E. Test Results Summary

Feature / Test Area	Metric	Result
Offline Core Features	All modules functional without internet	100% Success
RBAC Enforcement	Cross-role access attempts blocked	100% Accuracy
Medicine Notifications	On-time delivery on physical device	100% Accuracy
Pregnancy Week Calculation	Date accuracy vs. expected due date	Within ± 1 Day
SQLite Read Latency	Average dashboard query time	< 50 ms
SQLite Write Latency	Average write operation time	< 100 ms
MediBot Response Quality	Relevant responses for common queries	High
App Stability / Uptime	During extended testing sessions	99%
Notification (Background)	Delivery when app is closed	Confirmed (Android 10, 12)

Table 5: Summary of Test Results

F. User Experience

Representative users from both roles — pregnant women and doctors — evaluated the application through structured user acceptance testing sessions. Feedback consistently highlighted the application's offline reliability, intuitive navigation, and the practical value of the medicine reminder notification system. The MediBot chatbot received positive feedback for providing accessible health guidance without requiring internet access. Overall user satisfaction was rated as high across all evaluated modules.

VI. DISCUSSION

The results confirm that Pregamaa successfully delivers a reliable, offline-first maternal healthcare solution that addresses the critical gaps identified in the literature review. The application's Clean Architecture foundation ensures that each layer remains independently testable and maintainable, while the Riverpod state management layer provides reactive, efficient UI updates without performance degradation.

The 100% offline functionality achievement validates the architectural decision to use SQLite as the primary data store over a cloud-first approach. The sub-50ms database read latency demonstrates that local SQLite operations are performant enough for real-time UI interactions, even on mid-range Android devices.

The MediBot offline chatbot, while currently based on pre-trained response logic, demonstrates the feasibility of offline AI assistance in resource-constrained healthcare settings. Its integration within the unified application — alongside pregnancy tracking, reminders, and report management — eliminates the need for pregnant women to navigate multiple disconnected applications.

Limitations: The current version of Pregamaa does not support real-time cloud synchronization, live video consultations, or multi-device data access. The MediBot chatbot's offline response quality is constrained by its pre-trained dataset. These limitations are acknowledged and addressed in the future scope section.

VII. FUTURE SCOPE

Pregamaa's modular Clean Architecture provides a strong foundation for the following planned enhancements:

- **Cloud Backend Integration:** A REST API backend (Node.js/Django) with PostgreSQL/MongoDB enabling cross-device synchronization while maintaining offline-first capability via a bi-directional sync engine.
- **Real-Time Telemedicine:** WebSocket/WebRTC-based live chat and video consultation between pregnant women and doctors.
- **LLM-Powered MediBot:** Upgrade MediBot with a large language model (LLM) API for dynamic, context-aware maternal health guidance beyond the current pre-trained dataset.
- **AI-Based Predictive Analytics:** Machine learning models for personalized health recommendations and early high-risk pregnancy identification based on individual health patterns.
- **Wearable Device Integration:** IoT connectivity with smartwatches and pregnancy monitoring bands for automated vital sign tracking.
- **Multi-Language Support:** Hindi, Marathi, and other regional Indian languages to improve rural accessibility.
- **Government Health Scheme Integration:** Links to national maternal health programmes and government benefit schemes.
- **Biometric Authentication:** Fingerprint and face recognition for enhanced application security.

VIII. CONCLUSION

This paper has presented Pregamaa, an offline-first maternal healthcare mobile application developed using Flutter and Clean Architecture principles. The system successfully delivers a comprehensive, role-based pregnancy care platform — combining pregnancy tracking, medicine reminders, food guidance, medical report management, health articles, and AI chatbot assistance — that operates reliably without internet connectivity.

Experimental evaluation confirmed 100% offline functionality, 100% role-based access control accuracy, sub-50ms database response times, and 100% on-device notification delivery accuracy. User acceptance testing demonstrated high satisfaction across both user roles, particularly for the offline reliability and medicine reminder features.

Pregamaa addresses a critical and underserved need in maternal healthcare technology: a unified, offline-capable mobile platform accessible to pregnant women in low-connectivity environments. By establishing a scalable, modular architecture ready for future cloud and AI integration, Pregamaa represents a meaningful and extensible contribution to

digital maternal healthcare in developing regions.

ACKNOWLEDGMENT

The authors express sincere gratitude to [Guide Name], Department of Computer Science and Engineering, Government College of Engineering, Chandrapur, for invaluable guidance and mentorship throughout this research work. The authors also thank the Head of Department, [HOD Name], and Principal, [Principal Name], for providing institutional support and resources.

REFERENCES

- [1] S. Lund et al., "Mobile health intervention reduces perinatal mortality in Zanzibar: secondary outcomes of a cluster randomized controlled trial," *BJOG: An International Journal of Obstetrics & Gynaecology*, 2016.
- [2] J. E. Hirst et al., "Feasibility and acceptability of a smartphone-based application to monitor maternal well-being in rural India," *Pilot and Feasibility Studies*, 2018.
- [3] G. Ledger et al., "A smartphone application for pregnancy tracking and symptom logging," *Journal of Mobile Technology in Medicine*, 2014.
- [4] T. W. Bickmore et al., "Engagement vs. Deceit: Virtual Agents and User Wellbeing," in *Intelligent Virtual Agents*, Springer, 2009.
- [5] L. Laranjo et al., "Conversational agents in healthcare: a systematic review," *Journal of the American Medical Informatics Association*, vol. 25, no. 9, pp. 1248-1258, 2018.
- [6] C. Hartung et al., "Open Data Kit: Tools to Build Information Services for Developing Regions," in *Proc. 4th ACM/IEEE Int. Conf. on ICT and Development*, 2010.
- [7] W. Brunette et al., "Open Data Kit Sensors: A Sensor Integration Framework for Android at the Application Level," in *Proc. MobiSys*, 2013.
- [8] R. C. Martin, *Clean Architecture: A Craftsman's Guide to Software Structure and Design*, Prentice Hall, 2017.
- [9] Google LLC, Flutter Documentation, <https://flutter.dev/docs>, Accessed 2025.
- [10] Riverpod Documentation, <https://riverpod.dev/docs>, Accessed 2025.
- [11] SQLite Documentation, <https://www.sqlite.org/docs.html>, Accessed 2025.
- [12] World Health Organization, Maternal Health, <https://www.who.int/health-topics/maternal-health>, Accessed 2025.
- [13] flutter_local_notifications Plugin, https://pub.dev/packages/flutter_local_notifications, Accessed 2025.
- [14] Google LLC, Riverpod State Management for Flutter, <https://pub.dev/packages/riverpod>, Accessed 2025.